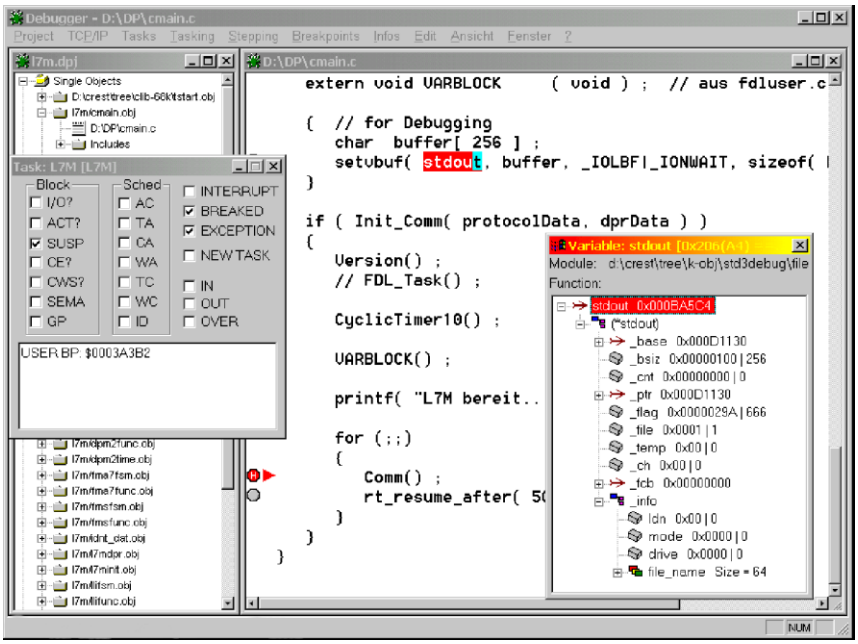
Ingenieurbüro für Echtzeitprogrammierung

RT-Debug

Quellcode-Debugging für ANSI-C und PEARL



Remote-Debugging

Remote-Debugging bietet den vollen Komfort einer grafischen Bedienoberfläche auch bei der Programmentwicklung für kleine Systeme. Die Trennung in leistungsfähige Bedienoberfläche und kleinen Debugger-Kern sichert das nahezu unbeeinflußte Programmverhalten auf dem Zielsystem.

Mit der Anbindung des Zielsystems über Standard-Netzwerkkomponenten wird sogar Fernüberwachung über das Internet möglich.

Source-Level

Die Untersuchung des Programmverhaltens erfolgt durchgängig auf Source-Level. Ob CREST-C oder UH-PEARL, RT-DEBUG bietet den Zugriff auf alle markanten Programmobjekte und berücksichtigt die Besonderheiten der einzelnen Sprachen.

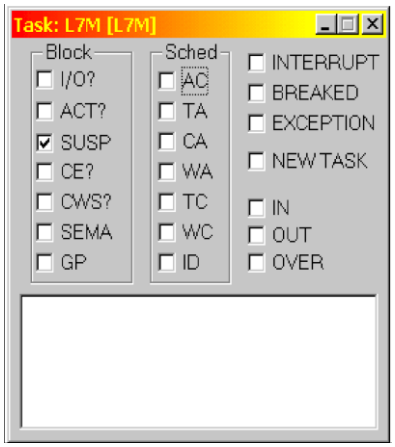
Der Programmablauf wird im Quelltext dargestellt, der Zugriff auf Variablen erfolgt typgerecht auch für anwenderdefinierte Datentypen. Die Besonderheiten des Multitasking beim Echtzeitsystem RTOS-UH stehen unter voller Kontrolle des Anwenders.

Crash-Analyse

Selbst bei katastrophalen Programmabbrüchen wie Bus- oder Address-Error bietet RT-Debug Unterstützung:

Aufruf-Rückverfolgung (Call-Stack/Backtrace) und Untersuchung der zum Crash-Zeitpunkt gültigen Variablenwerte klären die Ursache auch solch gravierender Fehler im Regelfall schnell. Im Notfall ist auch der Zugriff auf Assembler-Code, Registerinhalte und die Verwaltungsdaten einzelner Tasks möglich.

Ablaufkontrolle



RT-Debug zeigt den aktuellen Zustand der zu überwachenden Task in einem Zustandsfenster an.

Die kontinuierliche Verfolgung der Task-Zustände gibt einen präzisen Eindruck vom Laufverhalten. Besondere Ereignisse werden in einem eigenen Meldungsfenster protokolliert.

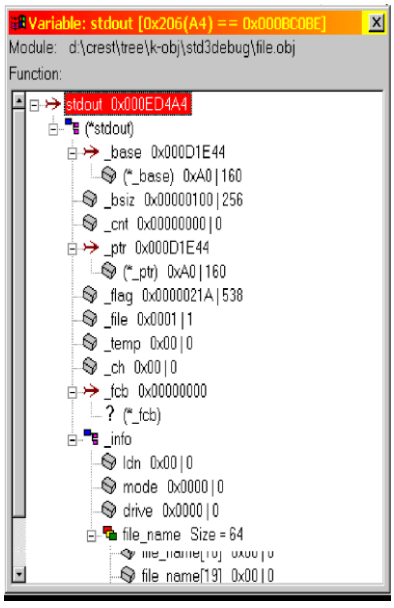
Eine Task kann zu jedem beliebigen Zeitpunkt in ihrem Ablauf unterbrochen werden. Die aktuelle Programmposition wird im Quelltextfenster grafisch dargestellt.

Zur genauen Kontrolle über den Programmablauf dient die überwachte Fortführung der Task mit den Befehlen:

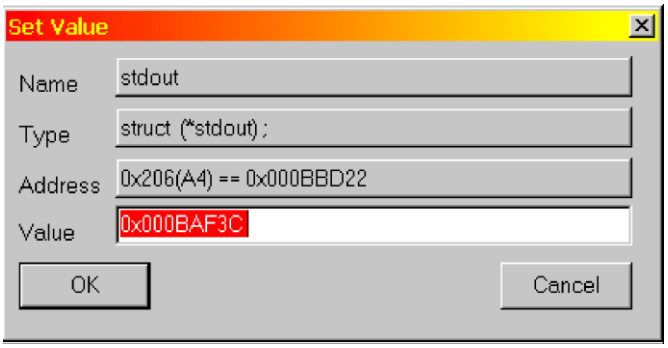
- Step-In – setzt das Programm im Einzelschritt fort
- Step-Out – stoppt die Task am Ende der aktuellen Prozedur
- Step-Over – stoppt die Task nach einem Prozeduraufruf

Breakpoints zur gezielten Programmunterbrechung an interessanten Programmstellen werden direkt im Quelltextfenster gesetzt.

Quick-Watch



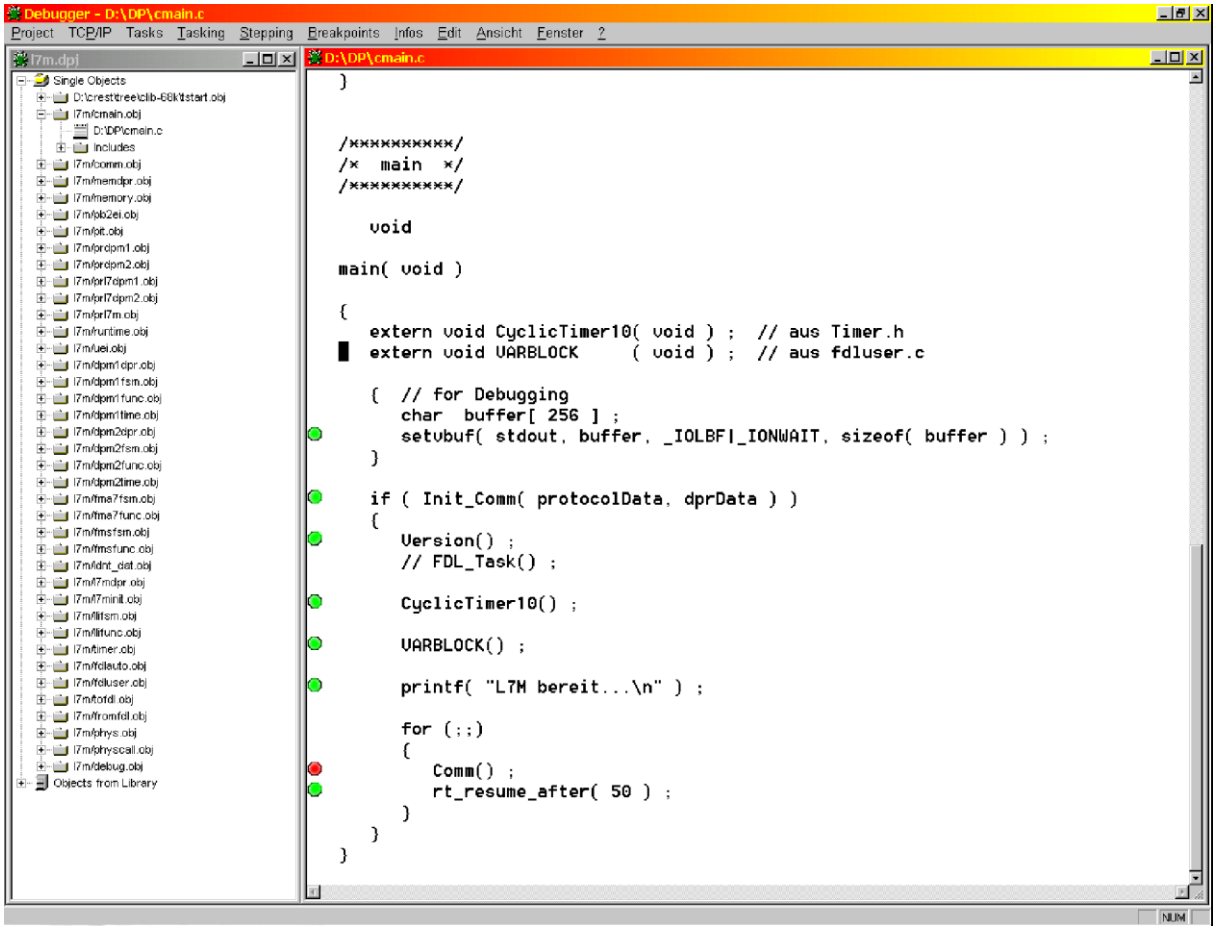
Die Werte aller lokalen und globalen Variablen werden in einem Variablenfenster angezeigt. Bei komplexen Datentypen bietet die aufklappbare Baumansicht schnellen Überblick und einfachen Zugriff auf Elementvariablen. Die symbolische Kennzeichnung unterschiedlicher Datentypen erleichtert den Überblick auch bei größeren Projekten.



Ein Quick-Watch-Dialog zeigt alle verfügbaren Informationen über einzelne Variablen und ermöglicht die direkte Änderung der Werte.

Projekt Organisation

RT-Debug stellt alle verfügbaren Projektinformationen in übersichtlicher, hierarchischer Form dar. Die Abhängigkeiten der einzelnen Übersetzungseinheiten sowie der eingebundenen Bibliotheksroutinen werden visualisiert und erleichtern die Navigation durch den Quelltext.



Die einzelnen Programmquelltexte sind einfach und schnell durch Navigation im Projektbaum erreichbar. Die gleichzeitige Anzeige mehrerer Quelltextfenster bietet optimale Übersicht über den Programmablauf.

Aktive wie auch mögliche Breakpoints sind farblich gekennzeichnet, die aktuelle Position im Programmablauf wird hervorgehoben. Mit Breakpoints oder von asynchronen Programmunterbrechungen angelaufene Quelldateien werden automatisch geöffnet und in den Vordergrund geholt.

Die Bedienung erfolgt wahlweise mit Maus oder Tastatur.

Zielsysteme

RT-Debug unterstützt alle RTOS-UH-Systeme auf PowerPC und 68*xxx*-Basis. Ein kleiner Debuggerkern auf dem Zielsystem kommuniziert mit der komfortablen Bedienoberfläche über TCP/IP-Protokoll. Die Kopplung zwischen den Systemen erfolgt über serielle Schnittstelle oder Netzwerk.

Der Debugger-Kern stellt Debug-Grundfunktionen zur Verfügung. Neben der Manipulation von Speicherbereichen setzt oder löscht er Breakpoints, überwacht Taskzustände und erkennt besondere Ereignisse im Taskablauf. Alle Aktionen werden vom Entwicklungsrechner angestoßen.

Die Aufteilung des Debuggers in einen Kern mit elementaren Grundfunktionen und eine komfortable Bedienoberfläche führt zu sehr geringer Belastung des Zielsystems. Das Zielsystem muß keine besonderen Anforderungen hinsichtlich verfügbarer Speichergröße oder Rechenkapazität erfüllen. Selbst Programme für kompakte Zielsysteme mit wenig verfügbarem Speicher können ohne spezielle Hardware-Hilfsmittel komfortabel entwickelt werden.

Entwicklungssystem

Der Entwicklungsrechner stellt die Hauptfunktionalitäten des Debuggers zu Verfügung. Er übersetzt die Quelldateien, analysiert und interpretiert die Debug-Informationen der Compiler und stellt diese übersichtlich grafisch dar. Die Bedienerführung folgt dem Look-and-Feel des Betriebssystems.

Als Betriebssysteme für den Entwicklungsrechner werden alle 32-Bit Microsoft-Windows Betriebssysteme seit Windows 95 auf Standard-PCs unterstützt.

Durch die Ankopplung der Zielsysteme über Netzwerk ist die örtliche Trennung zwischen Arbeitsplatz und Zielsystem möglich. Debug-Sitzungen können selbst über das Internet hinweg durchgeführt werden.